

# HPCS Application Analysis and Assessment

Dr. Jeremy Kepner  
MIT Lincoln Laboratory  
[kepner@ll.mit.edu](mailto:kepner@ll.mit.edu)

Dr. David Koester  
The MITRE Corporation  
[dkoester@mitre.org](mailto:dkoester@mitre.org)

## *Abstract*

**The value of a HPC system to a user includes many factors, such as: execution time on a particular problem, software development time, and both direct and indirect costs. The DARPA High Productivity Computing Systems is focused on providing a new generation of economically viable high productivity computing systems for the national security and industrial user community in the 2007-2010 timeframe. The goal is to provide systems that double in productivity (or value) every 18 months. This program has initiated a fundamental reassessment of how we define and measure performance, programmability, portability, robustness and ultimately productivity in the HPC domain. This talk will describe the HPCS efforts to develop a productivity assessment framework (see Figure 1), characterize HPC user workflows, and define the scope of the target applications.**

## **Introduction**

The HPCS program seeks to create trans-Pefaflop systems of significant value to the Government HPC community. Such value will be determined by assessing many additional factors beyond just theoretical peak flops (i.e. “Machoflops”). Ultimately, the goal is to decrease the time-to-solution, which means decreasing both the execution time and development time of an application on a particular system. Evaluating the capabilities of a system with respect to these goals requires a different assessment process. The goal of the HPCS assessment activity is to prototype and baseline a process that can be transitioned to the acquisition community for 2010 procurements.

## **Development Time**

The most novel part of the assessment activity will be the effort to measure/predict the ease or difficulty of developing HPC applications. Currently, there is no quantitative methodology for comparing the development time impact of various HPC programming technologies. To achieve this goal, we will use a variety of tools including

- Application of code metrics on existing HPC codes
- Several prototype analytic models of development time
- Interface characterization (e.g. language, parallel model, memory model, ...)
- Scalable benchmarks designed for testing both performance and programmability
- Classroom software engineering experiments
- Human validated demonstrations

These tools will provide the baseline data necessary for modeling development time and allow the new technologies developed under HPCS to be assessed quantitatively.

## Report Documentation Page

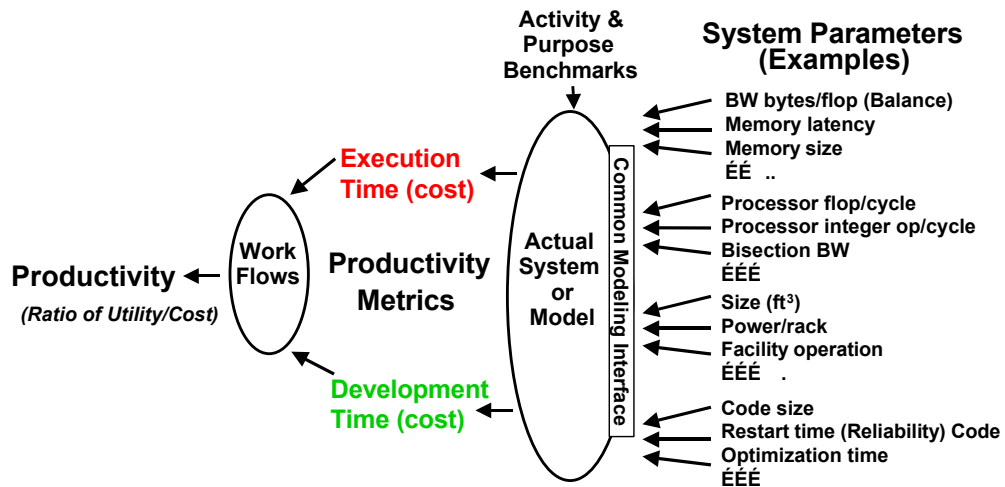
*Form Approved*  
*OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE <b>23 SEP 2003</b>	2. REPORT TYPE <b>N/A</b>	3. DATES COVERED <b>-</b>	
4. TITLE AND SUBTITLE <b>HPCS Application Analysis and Assessment</b>		5a. CONTRACT NUMBER	
		5b. GRANT NUMBER	
		5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)		5d. PROJECT NUMBER	
		5e. TASK NUMBER	
		5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>MIT Lincoln Laboratory; The MITRE Corporation</b>		8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)	
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release, distribution unlimited</b>			
13. SUPPLEMENTARY NOTES <b>See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop(7th). , The original document contains color images.</b>			
14. ABSTRACT			
15. SUBJECT TERMS			
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>	<b>UU</b>
			18. NUMBER OF PAGES <b>35</b>
			19a. NAME OF RESPONSIBLE PERSON

## Execution Time

The execution time part of the assessment activity will leverage the strong heritage in the HPC performance modeling community. This will include analytic, source code, and executable based tools for analyzing the projected performance of various applications on current, next generation and HPCS designs. The execution time and development time activities will be strongly coupled so as to provide a clear picture to the community of the tradeoffs that exist between execution time and development time.



**Figure 1: HPCS Assessment Framework.** The goal of the framework is to provide a mechanism for integrating system specific capabilities with user specific needs to assess the value of a particular machine for a particular mission.



# HPCS Application Analysis and Assessment

**Dr. Jeremy Kepner / Lincoln**

**Dr. David Koester / MITRE**

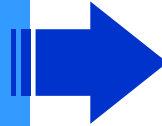
This work is sponsored by the Department of Defense under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.



# Outline



- **Introduction**



- *Motivation*
- *Productivity Framework*

- Workflows

- Metrics

- Models & Benchmarks

- Schedule and Summary



# High Productivity Computing Systems

## -Program Overview-

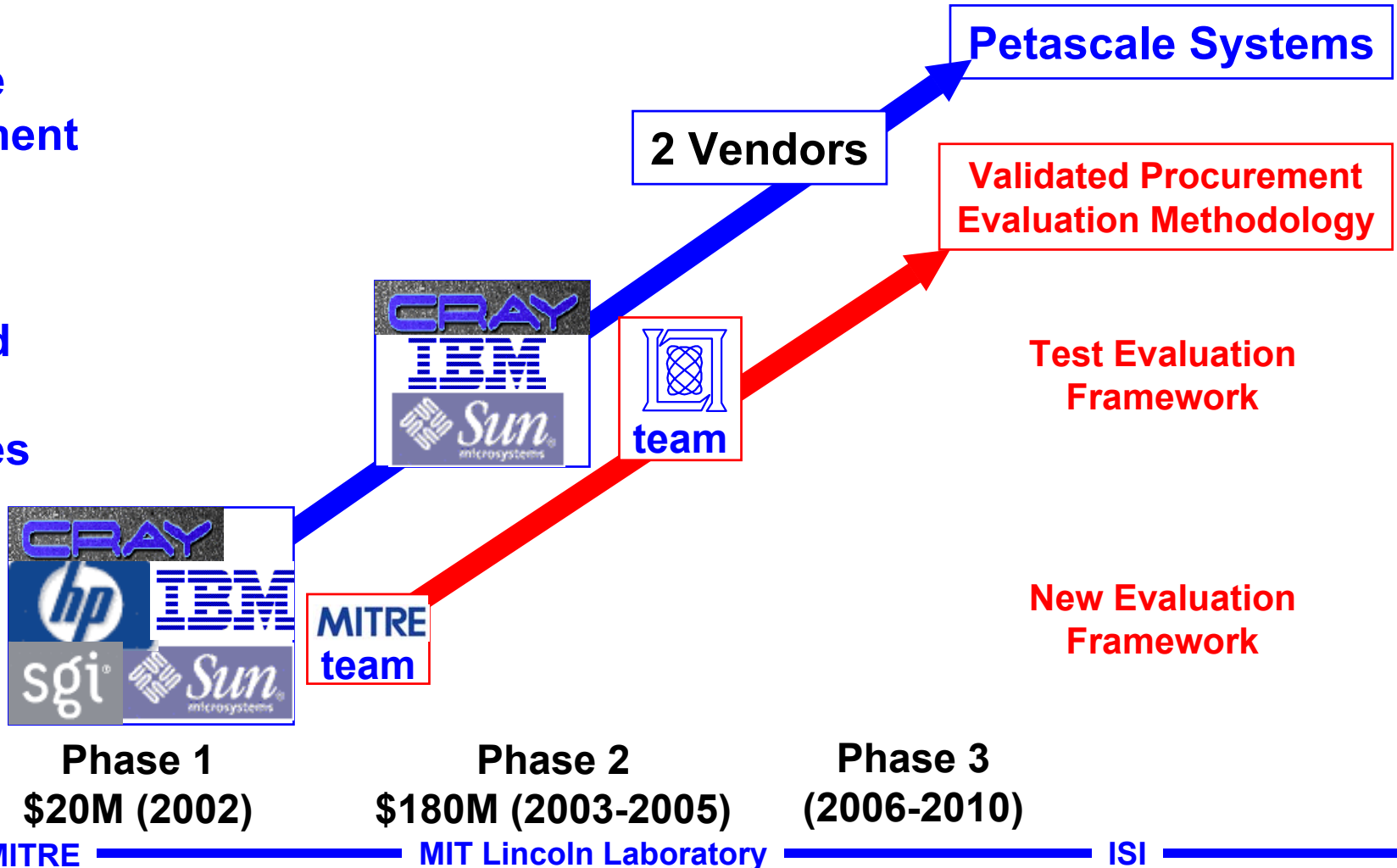


➤ Create a new generation of **economically viable computing systems** and a **procurement methodology** for the security/industrial community (2007 – 2010)

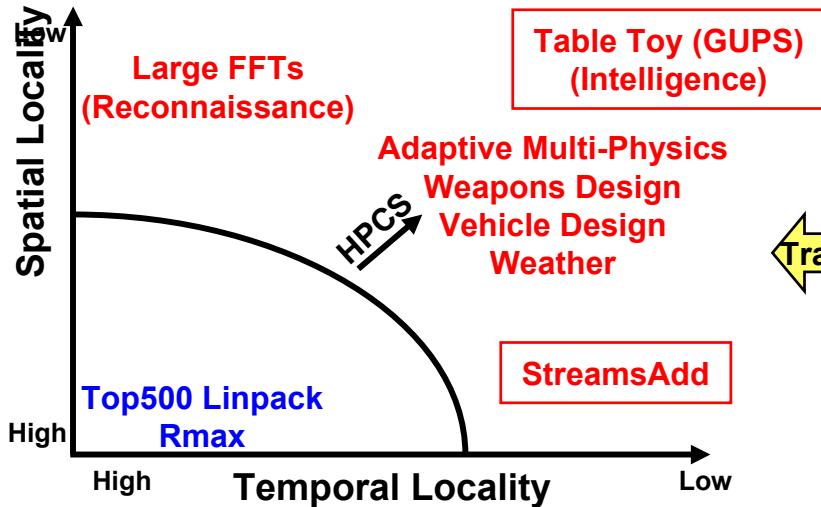
Full Scale  
Development

Advanced  
Design &  
Prototypes

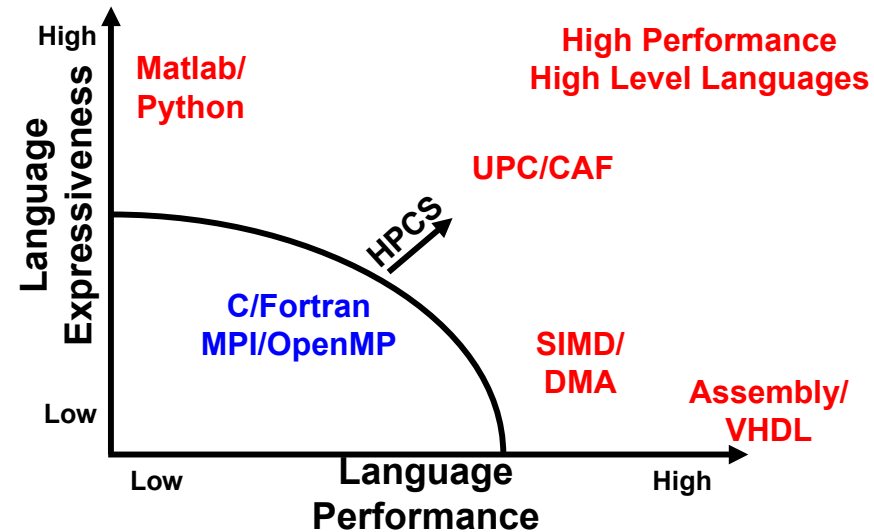
Concept  
Study



## Execution Time (Example)



## Development Time (Example)



Current metrics favor caches and pipelines

- Systems ill-suited to applications with
- Low spatial locality
- Low temporal locality

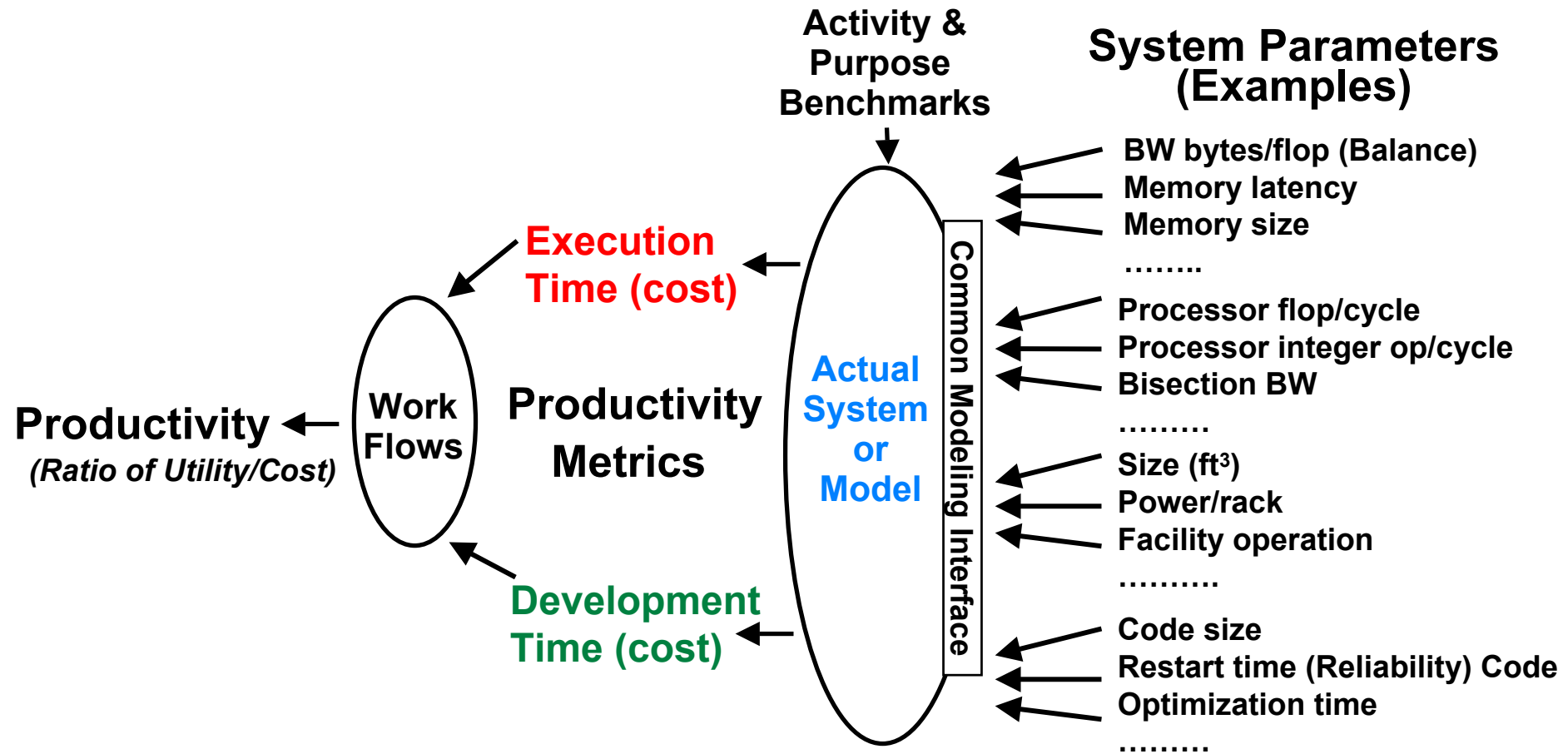
No metrics widely used

- Least common denominator standards
- Difficult to use
- Difficult to optimize

- HPCS needs a validated assessment methodology that values the “right” vendor innovations
- Allow tradeoffs between Execution and Development Time



# Phase 1: Productivity Framework





# Phase 2: Implementation



(Mitre, ISI, LBL, Lincoln, HPCMO, LANL & Mission Partners)

(Lincoln, OSU, CodeSourcery)

Performance Analysis  
(ISI, LLNL & UCSD)

Activity & Purpose  
Benchmarks

System Parameters  
(Examples)

BW bytes/flop (Balance)  
Memory latency  
Memory size  
.....

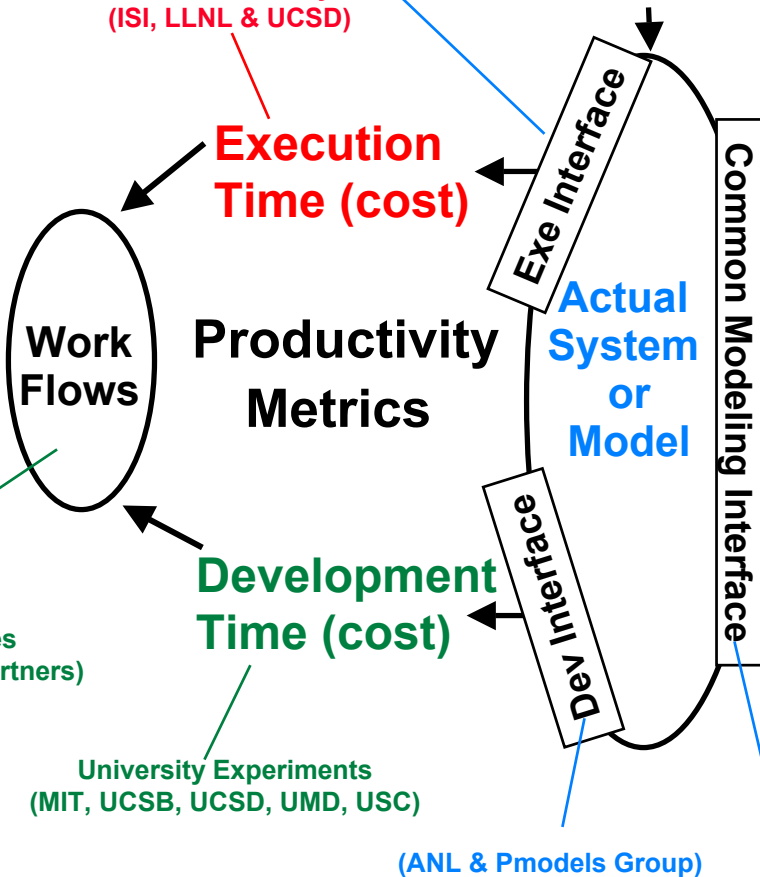
Processor flop/cycle  
Processor integer op/cycle  
Bisection BW  
.....

Size (ft<sup>3</sup>)  
Power/rack  
Facility operation  
.....

Code size  
Restart time (Reliability) Code  
Optimization time  
.....

(ISI, LLNL & UCSD)

(ANL & Pmodels Group)



Execution Time (cost)

Productivity Metrics

Development Time (cost)

Productivity  
(Ratio of Utility/Cost)

Metrics Analysis of  
Current and New Codes  
(Lincoln, UMD & Mission Partners)

University Experiments  
(MIT, UCSB, UCSD, UMD, USC)

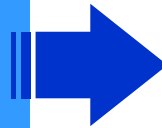


# Outline



- Introduction

- **Workflows**



- *Lone Researcher*
- *Enterprise*
- *Production*

- Metrics

- Models & Benchmarks

- Schedule and Summary



# HPCS Mission Work Flows

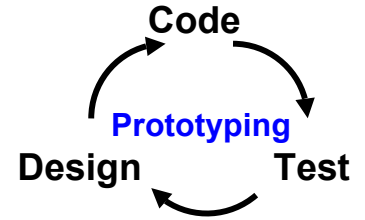
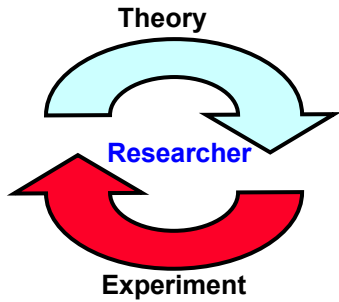


Overall Cycle      Development Cycle

**Researcher**

Days to hours

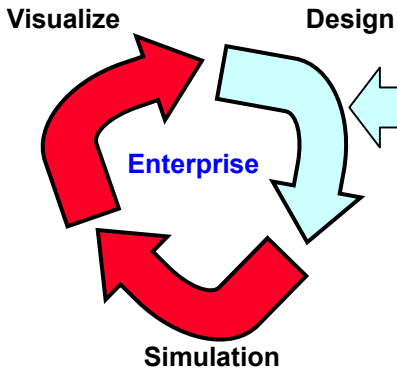
Hours to minutes



**Enterprise**

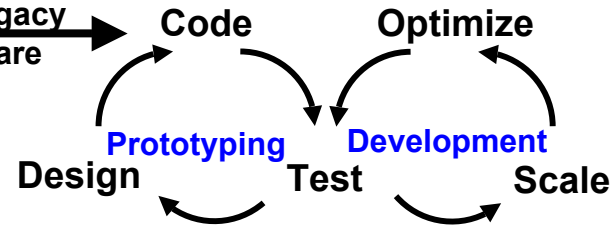
Months to days

Months to days



Port Legacy Software

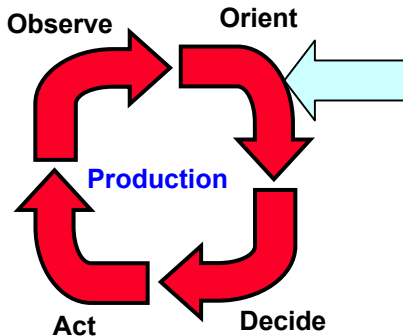
Port Legacy Software



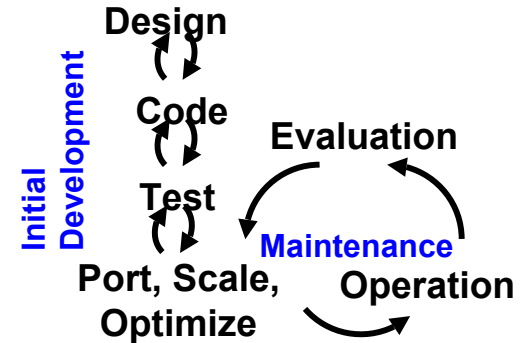
**Production**

Hours to Minutes (Response Time)

Years to months



Initial Product Development



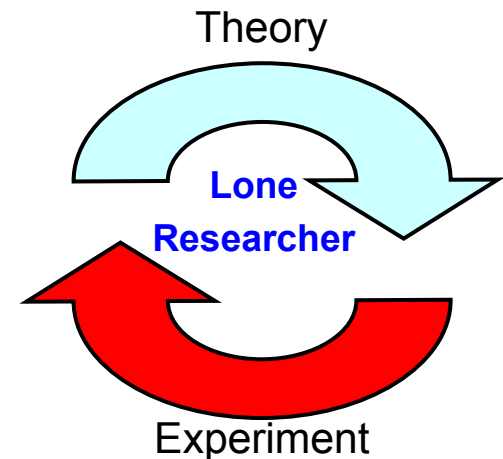
**HPCS Productivity Factors: Performance, Programmability, Portability, and Robustness are very closely coupled with each work flow**



# Lone Researcher



- **Missions (development):** Cryptanalysis, Signal Processing, Weather, Electromagnetics
- **Process Overview**
  - Goal: solve a compute intensive domain problem: crack a code, incorporate new physics, refine a simulation, detect a target
  - Starting point: inherited software framework (~3,000 lines)
  - Modify framework to incorporate new data (~10% of code base)
  - Make algorithmic changes (~10% of code base); Test on data; Iterate
  - Progressively increase problem size until success
  - Deliver: code, test data, algorithm specification
- **Environment overview**
  - Duration: months                      Team size: 1
  - Machines: workstations (some clusters), HPC decreasing
  - Languages: FORTRAN, C → Matlab, Python
  - Libraries: math (external) and domain (internal)
- **Software productivity challenges**
  - Focus on rapid iteration cycle
  - Frameworks/libraries often serial

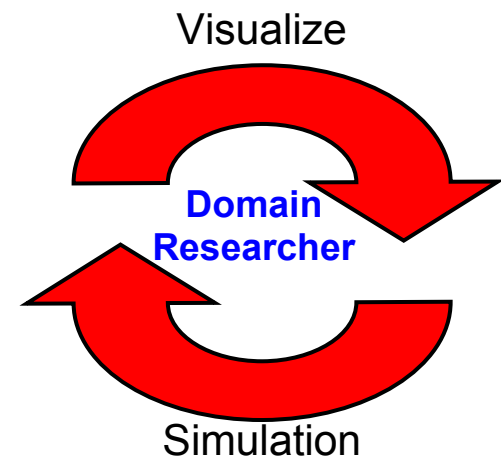




# Domain Researcher (special case)



- **Scientific Research: DoD HPCMP Challenge Problems, NNSA/ASCI Milestone Simulations**
- **Process Overview**
  - Goal: Use HPC to perform Domain Research
  - Starting point: Running code, possibly from an Independent Software Vendor (ISV)
  - NO modifications to codes
  - Repeatedly run the application with user defined optimization
- **Environment overview**
  - Duration: months                      Team size: 1-5
  - Machines: workstations (some clusters), HPC
  - Languages: FORTRAN, C
  - Libraries: math (external) and domain (internal)
- **Software productivity challenges — None!**
- **Productivity challenges**
  - Robustness (reliability)
  - Performance
  - Resource center operability



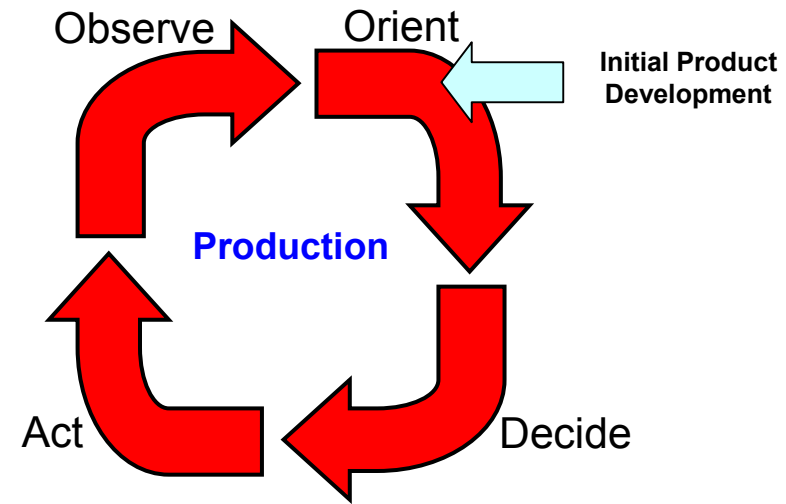


- Missions (production): Cryptanalysis, Sensor Processing, Weather
- Process Overview
  - Goal: develop a system for fielded deployment on an HPC system
  - Starting point: algorithm specification, test code, test data, development software framework
  - Rewrite test code into development framework; Test on data; Iterate
  - Port to HPC; Scale; Optimize (incorporate machine specific features)
  - Progressively increase problem size until success
  - Deliver: system

- Environment overview
  - Duration: ~1 year
  - Machines: workstations and HPC target
  - Languages: FORTRAN, C, → C++

Team size: 2-20

- Software productivity challenges
  - Conversion of higher level languages
  - Parallelization of serial library functions
  - Parallelization of algorithm
  - Sizing of HPC target machine





# HPC Workflow SW Technologies

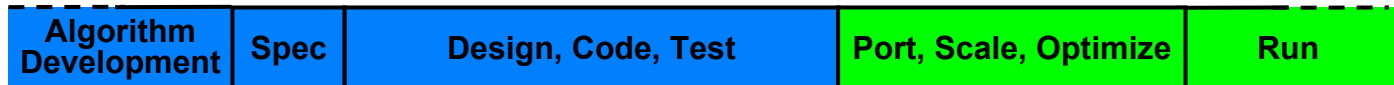


## Production Workflow

- Many technologies targeting specific pieces of workflow
- Need to quantify workflows (stages and % time spent)
- Need to measure technology impact on stages

Workstation

Supercomputer



Operating Systems		Linux			RT Linux		
Compilers	Matlab	Java	C++	OpenMP	F90	UPC Coarray	
Libraries		CORBA		VSIPL   VSIPL++	MPI	DRI	ATLAS, BLAS, FFTW, PETE, PAPI
Tools	UML		Globus	TotalView			
Problem Solving Environments		CCA		ESMF		POOMA	PVL

Mainstream Software

HPC Software

MITRE

MIT Lincoln Laboratory

ISI



# Example: Coding vs. Testing



## Workflow Breakdown (NASA SEL)

	Analysis and Design	Coding and Auditing	Checkout and Test
Sage	39%	14%	47%
NTDS	30	20	50
Gemini	36	17	47
Saturn V	32	24	44
OS/360	33	17	50
TRW Survey	46	20	34

## Testing Techniques (UMD)

### Code Reading

Reading by Stepwise Abstraction

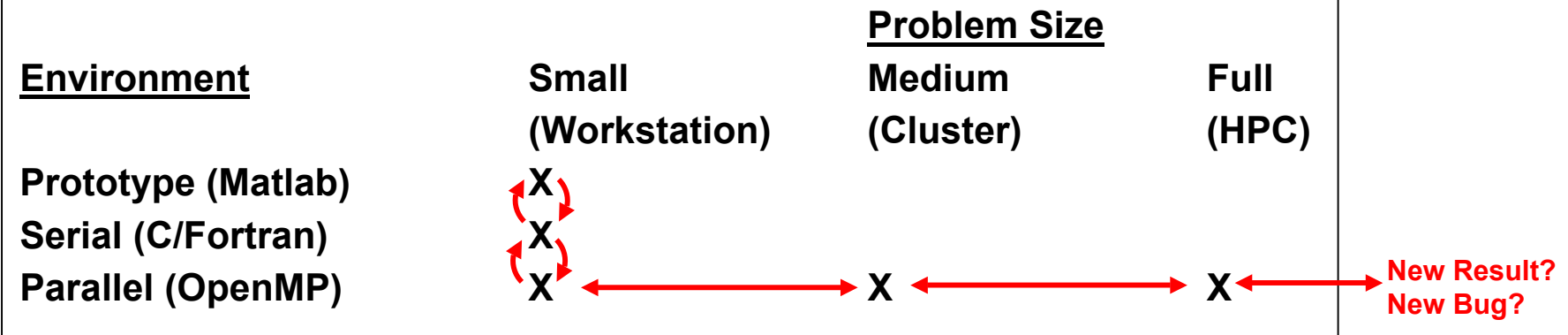
### Functional Testing

Boundary Value Equivalence Partition Testing

### Structural Testing

Achieving 100% statement coverage

## What is HPC testing process?



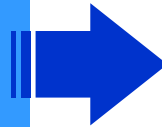


# Outline



- Introduction
- Workflows

- **Metrics**



- *Existing Metrics*
- *Dev. Time Experiments*
- *Novel Metrics*

- Models & Benchmarks
- Schedule and Summary



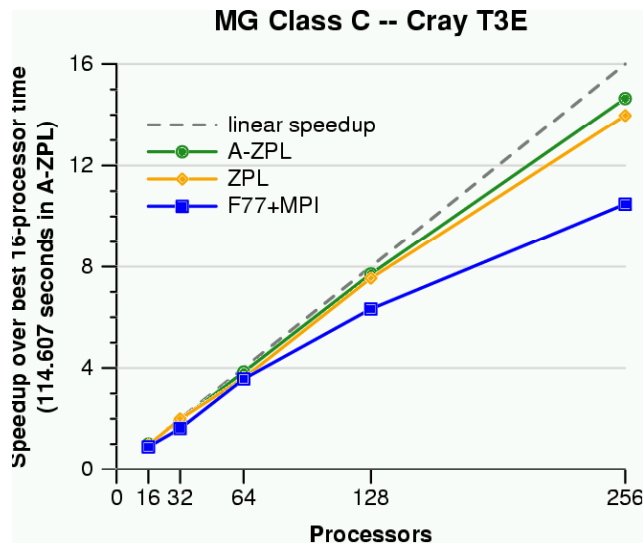
# Example Existing Code Analysis



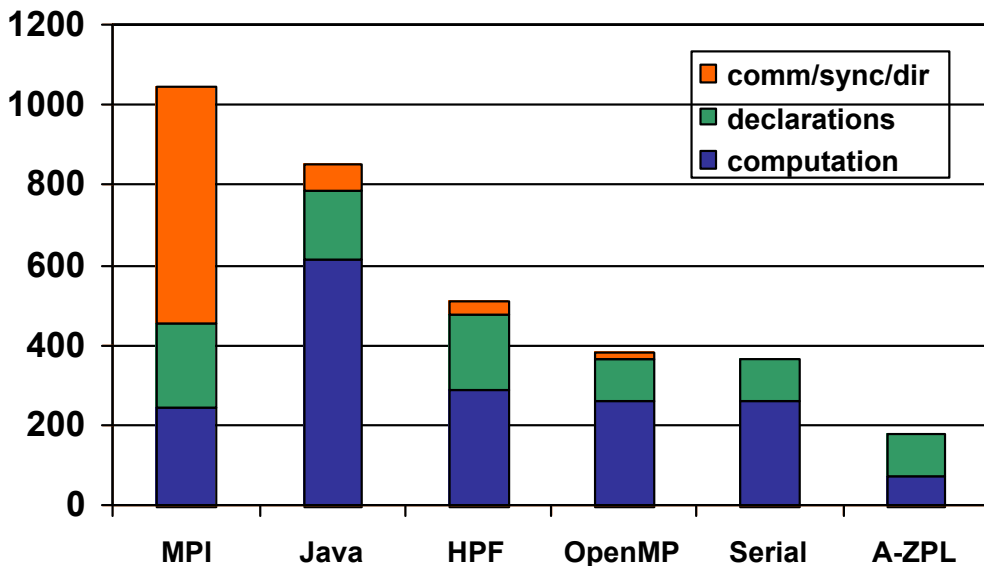
Analysis of existing codes used to test metrics and identify important trends in productivity and performance



## MG Performance



## NAS MG Linecounts



atory

ISI



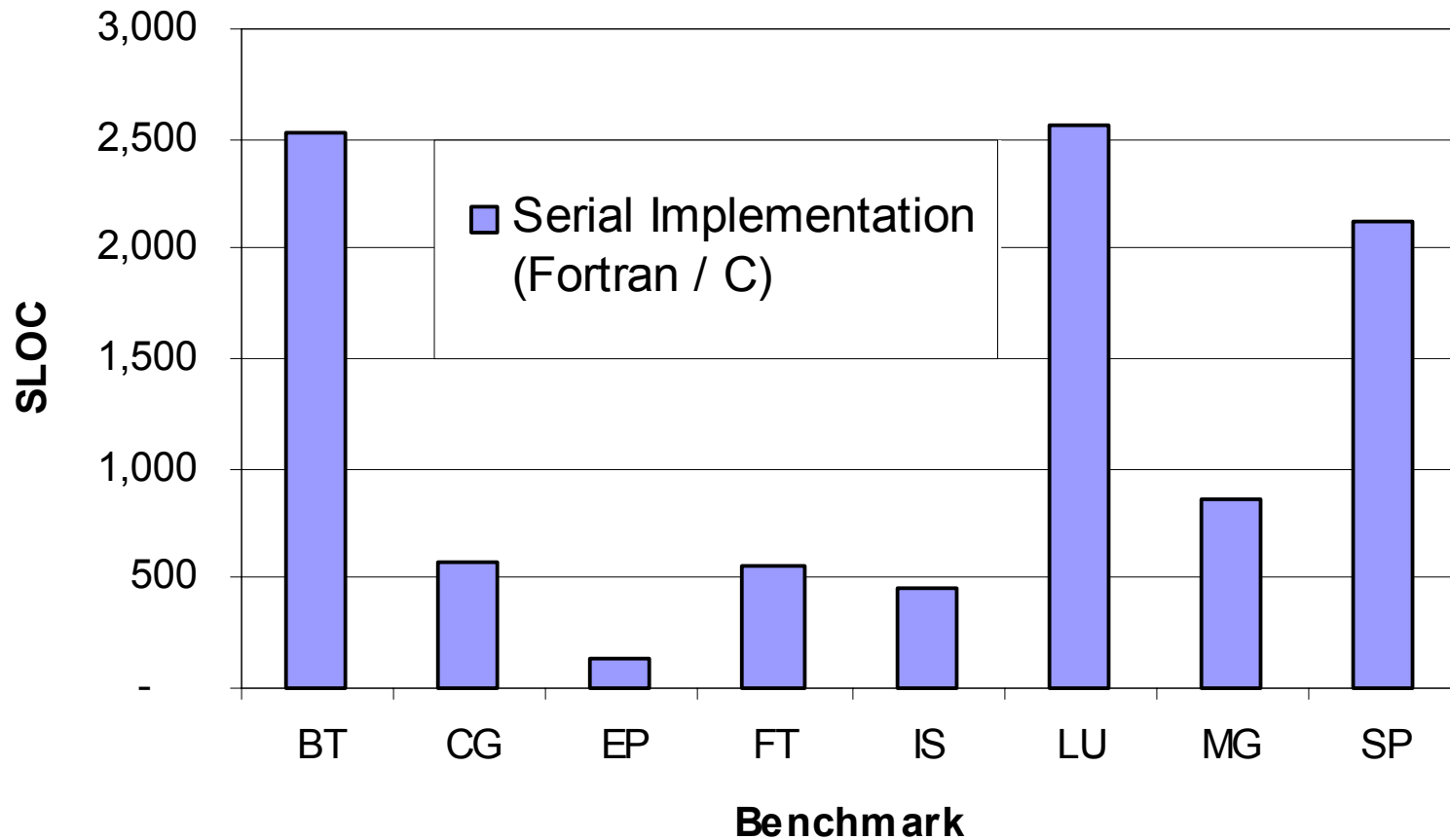
# NPB Implementations



Benchmark	Languages							
	Serial Fortran	Serial C	Fortran / MPI	C / MPI	Fortran / OpenMP	C / OpenMP	HPF	Java
BT	█		█		█		█	█
CG	█		█		█		█	█
EP	█		█		█			
FT	█		█		█		█	█
IS		█		█		█		█
LU	█		█		█		█	█
MG	█		█		█		█	█
SP	█		█		█		█	█

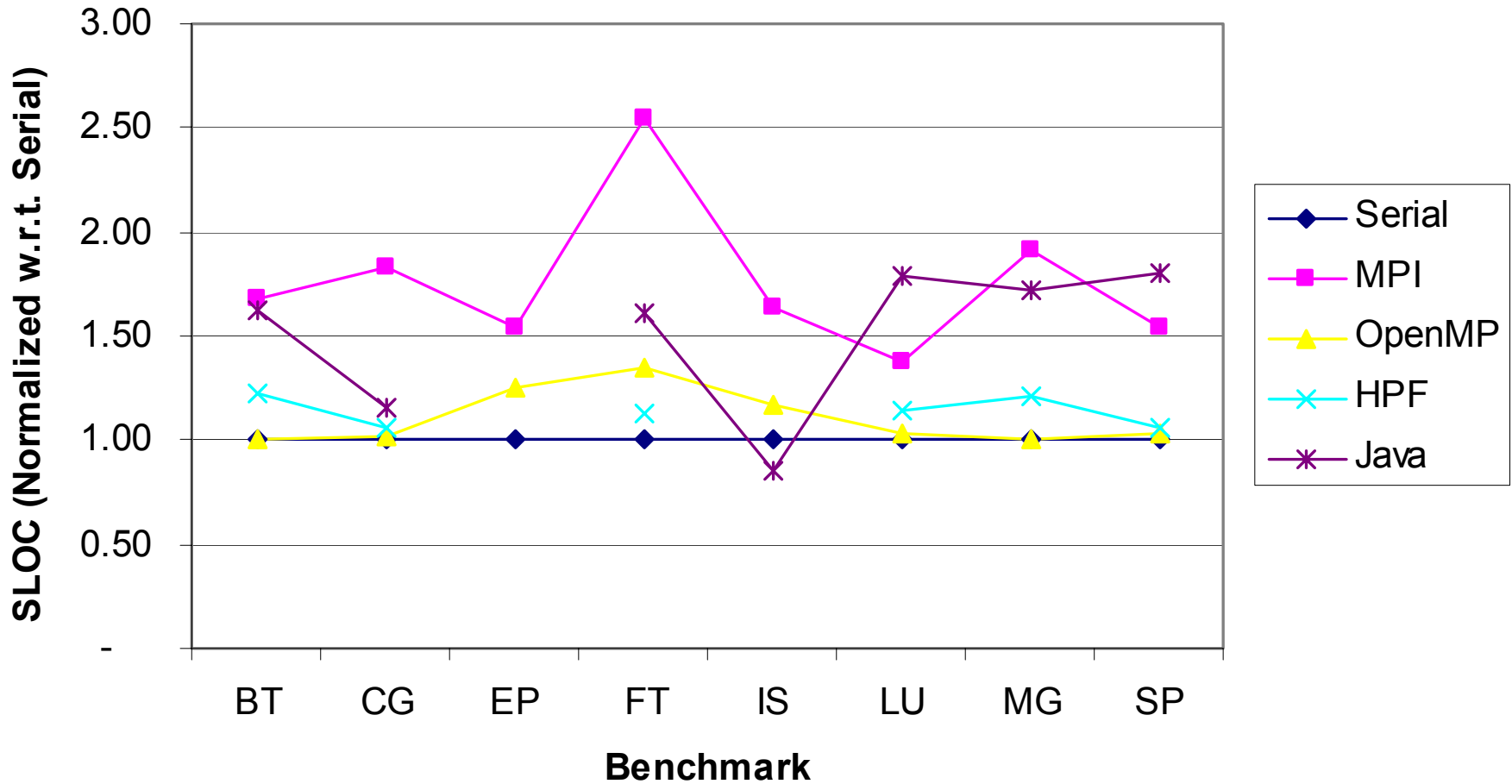


# Source Lines of Code (SLOC) for the NAS Parallel Benchmarks (NPB)



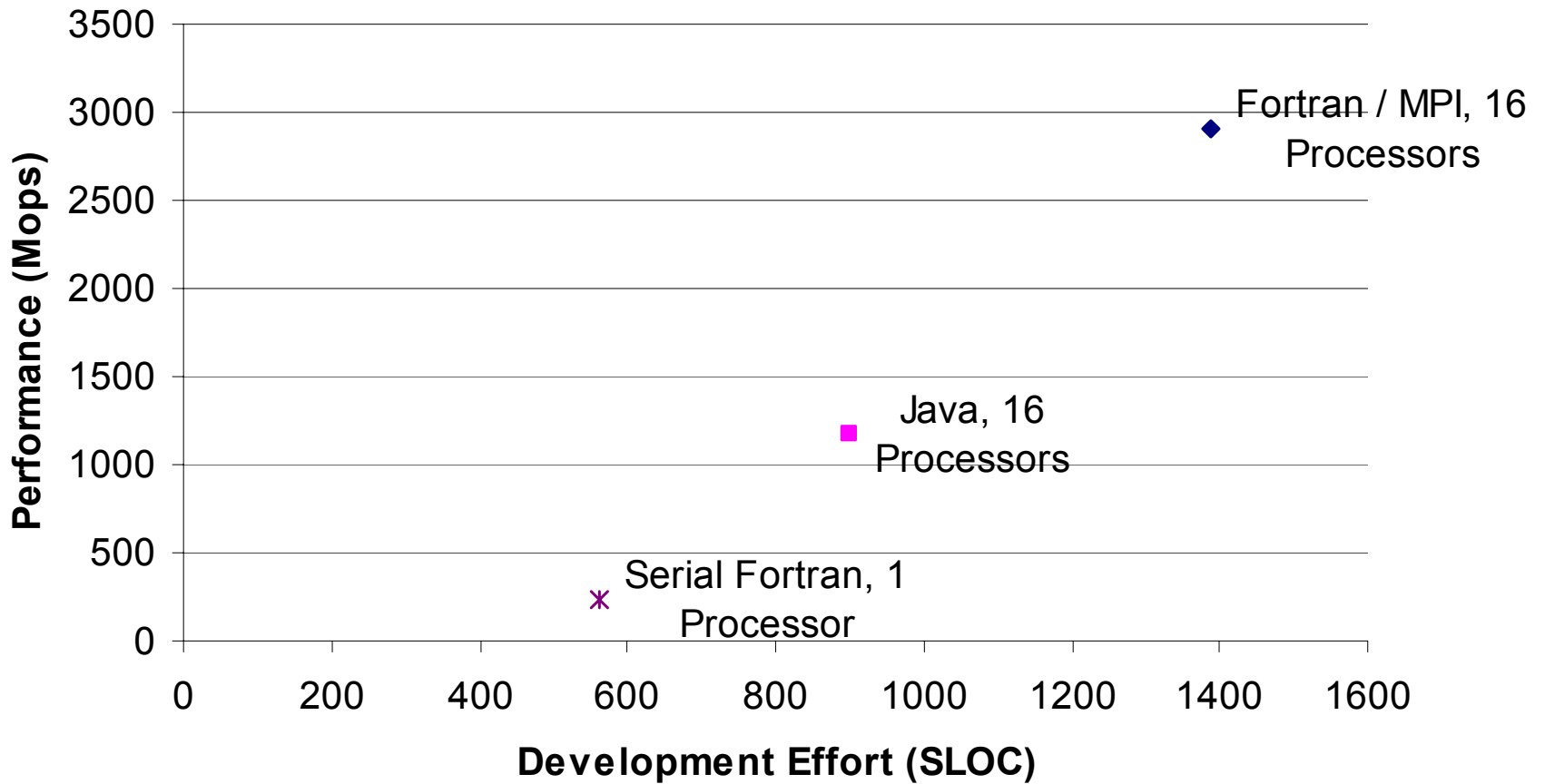


# Normalized SLOC for All Implementations of the NPB



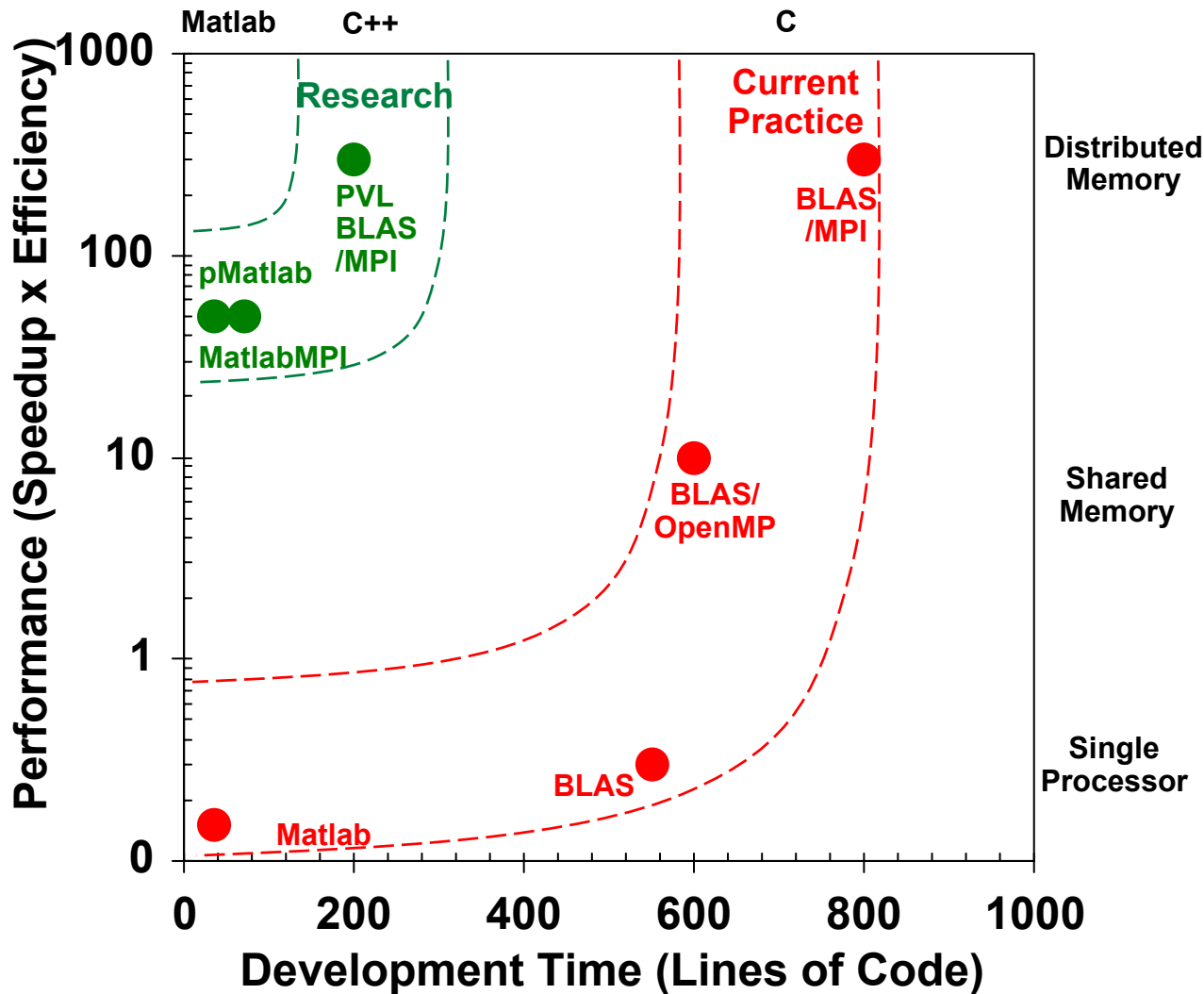


# NAS FT Performance vs. SLOCs





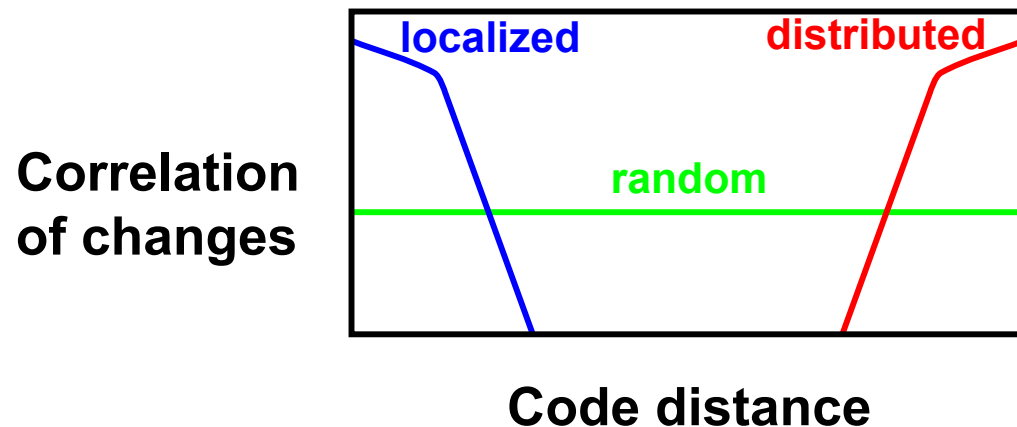
# Example Experiment Results (N=1)



- Same application (image filtering)
- Same programmer
- Different langs/libs
  - Matlab \*Estimate
  - BLAS
  - BLAS/OpenMP
  - BLAS/MPI\*
  - PVL/BLAS/MPI\*
  - MatlabMPI
  - pMatlab\*

Controlled experiments can potentially measure the impact of different technologies and quantify development time and execution time tradeoffs

- HPC Software Development often involves changing code ( $\Delta x$ ) to change performance ( $\Delta y$ )
  - 1st order size metrics measures scale of change  $E(\Delta x)$
  - 2nd order metrics would measure nature of change  $E(\Delta x^2)$
- Example: 2 Point Correlation Function
  - Looks at “distance” between code changes
  - Determines if changes are **localized (good)** or **distributed (bad)**

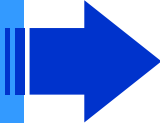


- Other Zany Metrics
  - See Cray talk



# Outline



- Introduction
- Workflows
- Metrics
- **Models & Benchmarks** 
  - *Prototype Models*
  - *A&P Benchmarks*
- Schedule and Summary



# Prototype Productivity Models



## Special Model with Work Estimator (Sterling)

$$\Psi_w = \frac{S_P \times E \times A}{c_f \times \left\{ \Gamma \times (\bar{\rho} \cdot \bar{n}) \right\} + (c_m + c_o) \times T}$$

## Utility (Snir)

$$P(S, A, U(.)) = \min_{\text{cost } t} \frac{U(T(S, A, \text{Cost}))}{\text{Cost}}$$

## Productivity Factor Based (Kepner)

$$\text{productivity}_{\text{Linpack}}^{\text{GUPS}} \approx \left( \frac{\left( \frac{\text{useful ops}}{\text{second}} \right)_{\text{Linpack}}^{\text{GUPS}}}{\text{Hardware Cost}} \right) \left( \text{productivity factor} \right) \left( \text{mission factor} \right)$$

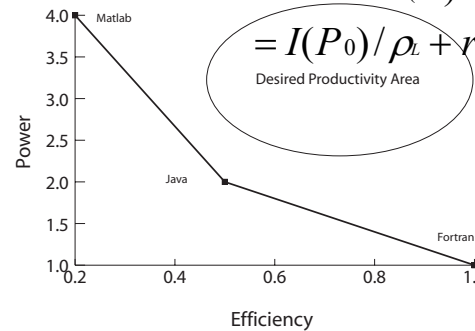
$$\left( \text{productivity factor} \right) \approx \left( \text{Language Level} \right) \times \left( \text{Parallel Model} \right) \times \text{Portability} \times \frac{\text{Availability}}{\text{Maintenance}}$$

## Efficiency and Power (Kennedy, Koelbel, Schreiber)

$$T(P_L) = I(P_L) + rE(P_L)$$

$$= I(P_0) \cdot \frac{I(P_L)}{I(P_0)} + rE(P_0) \cdot \frac{E(P_L)}{E(P_0)}$$

$$= I(P_0) / \rho_L + rE(P_0) / \varepsilon_L$$

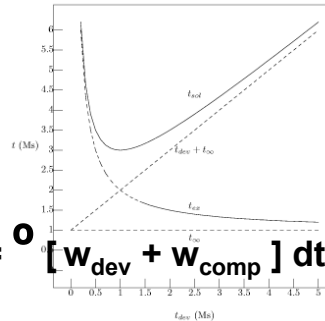


## CoCoMo II

(software engineering community)

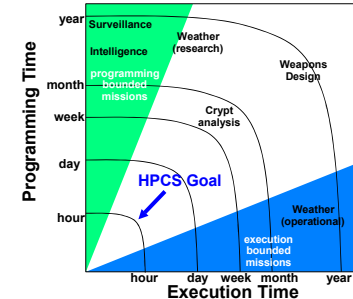
$$\left( \text{Effort Multipliers} \right) \times A \times \left( \text{Size} \right)^{\left( \text{Scale Factors} \right)}$$

## Least Action (Numrich)



$$S = 0 \quad \left[ w_{dev} + w_{comp} \right] dt; \quad \delta S = 0$$

## Time-To-Solution (Kogge)



**HPCS has triggered ground breaking activity in understanding HPC productivity**  
 -Community focused on *quantifiable* productivity (potential for broad impact)  
 -Numerous proposals provide a strong foundation for Phase 2



# Code Size and Reuse Cost



Lines of code  
**Function Points**  
**Reuse**  
 Re-engineering  
 Maintenance

$$\text{Code Size} = \left[ \text{New} \right] + \left[ \text{Reused} \right] + \left[ \text{Re-engineered} \right] + \left[ \text{Maintained} \right]$$

Measured in lines of code or functions points (converted to lines of code)

Lines per function point

C, Fortran	~100
Fortran77	~100
C++	~30
Java	~30
Matlab	~10
Python	~10
Spreadsheet	~5

## HPC Challenge Areas

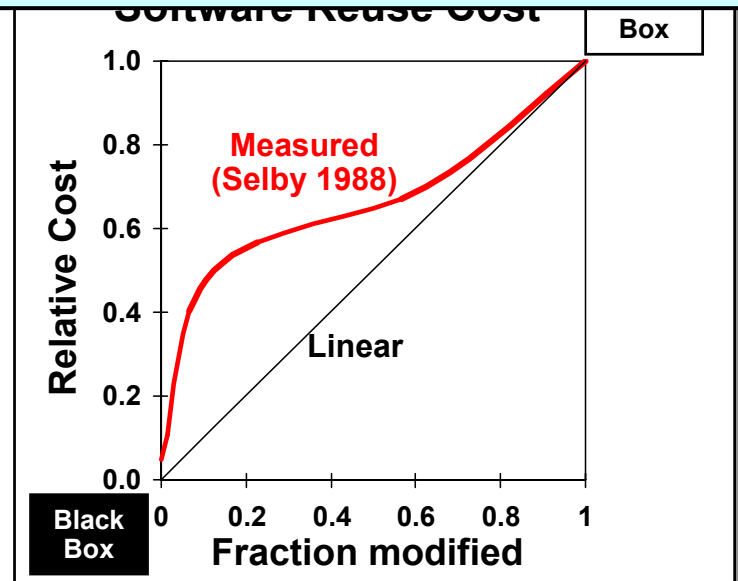
### Function Points

High productivity languages not available on HPC

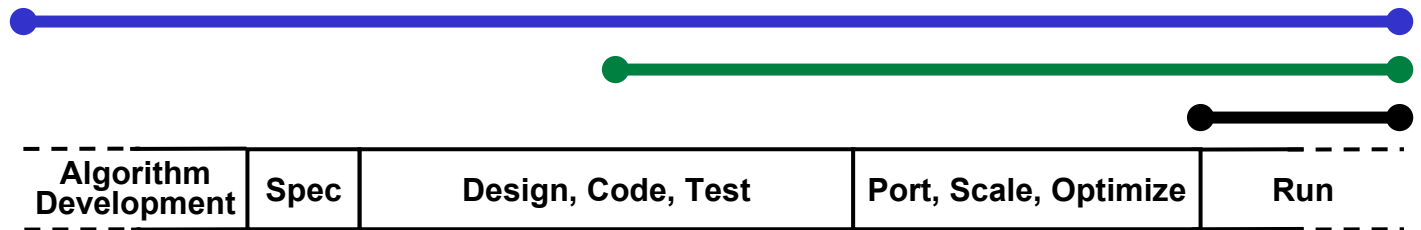
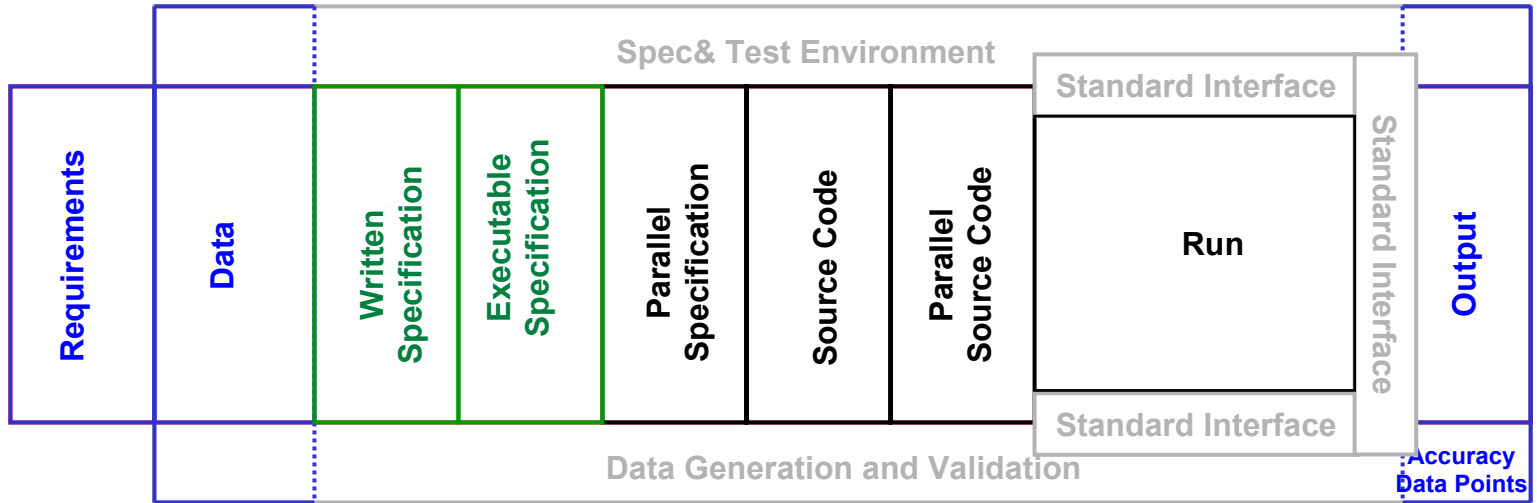
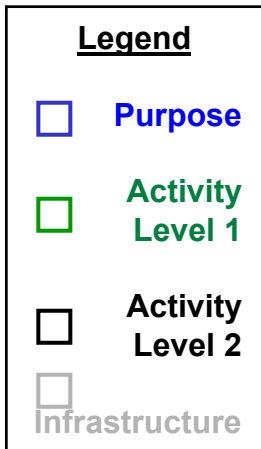
### Reuse

Nonlinear reuse effects. Performance requirements dictate “white box” reuse model

- Code size is the most important software productivity parameter
- Non-HPC world reduces code size by
  - Higher level languages
  - Reuse
- HPC performance requirements currently limit the exploitation of these approaches



## Activity & Purpose Benchmark



## Development Workflow

**Activity Benchmarks define a set of instructions (i.e., source code) to be executed**  
**Purpose Benchmarks define requirements, inputs and output**  
**Together they address the entire development workflow**



# HPCS Phase 1 Example Kernels and Applications



Mission Area	Kernels	Application	Source
Stockpile Stewardship	Random Memory Access	UMT2000	ASCI Purple Benchmarks
	Unstructured Grids		
	Eulerian Hydrocode Adaptive Mesh	SAGE3D	ASCI Purple Benchmarks
	Unstructured Finite Element Model Adaptive Mesh Refinement	ALEGRA	Sandia National Labs
Operational Weather and Ocean Forecasting	Finite Difference Model	NLOM	DoD HPCMP TI-03
Army Future Combat Weapons Systems	Finite Difference Model Adaptive Mesh Refinement	CTH	DoD HPCMP TI-03
Crashworthiness Simulations	Multiphysics Nonlinear Finite Element	LS-DYNA	Available to Vendors

Bio-Application	Kernels	Application	Source
Quantum and Molecular Mechanics	Macromolecular Dynamics	CHARMM	<a href="http://yuri.harvard.edu/">http://yuri.harvard.edu/</a>
	Energy Minimization		
	MonteCarlo Simulation		
Whole Genome Analysis	Sequence Comparison	Needleman- Wunsch	<a href="http://www.med.nyu.edu/rcr/rcr/course/sim-sw.html">http://www.med.nyu.edu/rcr/rcr/course/sim-sw.html</a>
		BLAST	<a href="http://www.ncbi.nlm.nih.gov/BLAST/">http://www.ncbi.nlm.nih.gov/BLAST/</a>
		FASTA	<a href="http://www.ebi.ac.uk/fasta33/">http://www.ebi.ac.uk/fasta33/</a>
		HMMR	<a href="http://hmmer.wustl.edu/">http://hmmer.wustl.edu/</a>
Systems Biology	Functional Genomics Biological Pathway Analysis	BioSpice (Arkin, 2001)	<a href="http://genomics.lbl.gov/~aparkin/Group/Codebase.html">http://genomics.lbl.gov/~aparkin/Group/Codebase.html</a>

Other Kernels	Kernels	Application	Source
	Lower / Upper Triangular Matrix Decomposition	LINPACK	Available on Web
	Conjugate Gradient Solver QR Decomposition		DoD HPCMP TI-03 Paper & Pencil for Kernels
	1D FFT		Paper & Pencil for Kernels
	2D FFT		Paper & Pencil for Kernels
	Table Toy (GUP/s)		Paper & Pencil for Kernels
	Multiple Precision Mathematics		Paper & Pencil for Kernels
	Dynamic Programming		Paper & Pencil for Kernels
	Matrix Transpose [Binary manipulation]		Paper & Pencil for Kernels
	Integer Sort [With large multiword key]		Paper & Pencil for Kernels
	Binary Equation Solution		Paper & Pencil for Kernels
	Graph Extraction (Breadth First) Search		Paper & Pencil for Kernels
	Sort a large set		Paper & Pencil for Kernels
	Construct a relationship graph based on proximity		Paper & Pencil for Kernels
	Various Convolutions		Paper & Pencil for Kernels
	Various Coordinate Transforms		Paper & Pencil for Kernels
	Various Block Data Transfers		Paper & Pencil for Kernels

**Set of scope benchmarks  
representing Mission Partner  
and emerging Bio-Science high-  
end computing requirements**



# Outline



- Introduction
- Workflows
- Metrics
- Models & Benchmarks
- **Schedule and Summary**



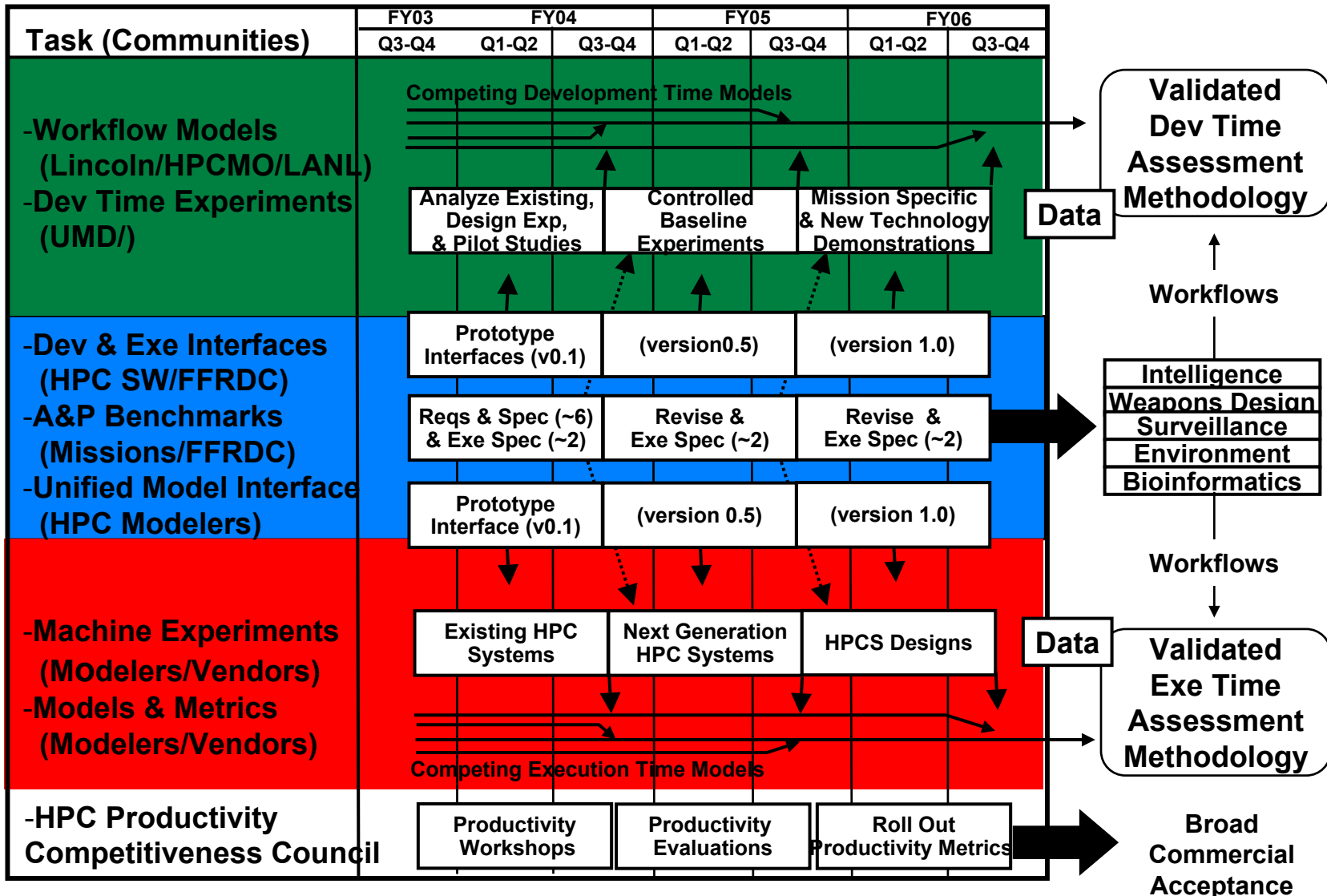
# Phase II Productivity Forum Tasks and Schedule



Development

Framework

Execution





# Summary



- **Goal is to develop an acquisition quality framework for HPC systems that includes**
  - **Development time**
  - **Execution time**
- **Have assembled a team that will develop models, analyze existing HPC codes, develop tools and conduct HPC development time and execution time experiments**
- **Measures of success**
  - **Acceptance by users, vendors and acquisition community**
  - **Quantitatively explain HPC rules of thumb:**
    - "OpenMP is easier than MPI, but doesn't scale a high"
    - "UPC/CAF is easier than OpenMP"
    - "Matlab is easier the Fortran, but isn't as fast"
  - **Predict impact of new technologies**



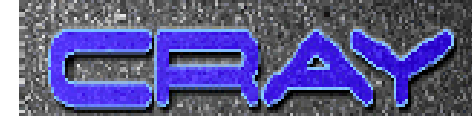
# Backup Slides



# HPCS Phase II Teams



## Industry:



PI: Elnozahy

PI: Gustafson

PI: Smith

## Goal:

- Provide a new generation of economically viable high productivity computing systems for the national security and industrial user community (2007 – 2010)

## Productivity Team (Lincoln Lead)



PI: Kepner

PI: Lucas

PI: Basili

PI: Benson & Snavelly



PI: Koester

PIs: Vetter, Lusk, Post, Bailey

PIs: Gilbert, Edelman, Ahalt, Mitchell

## Goal:

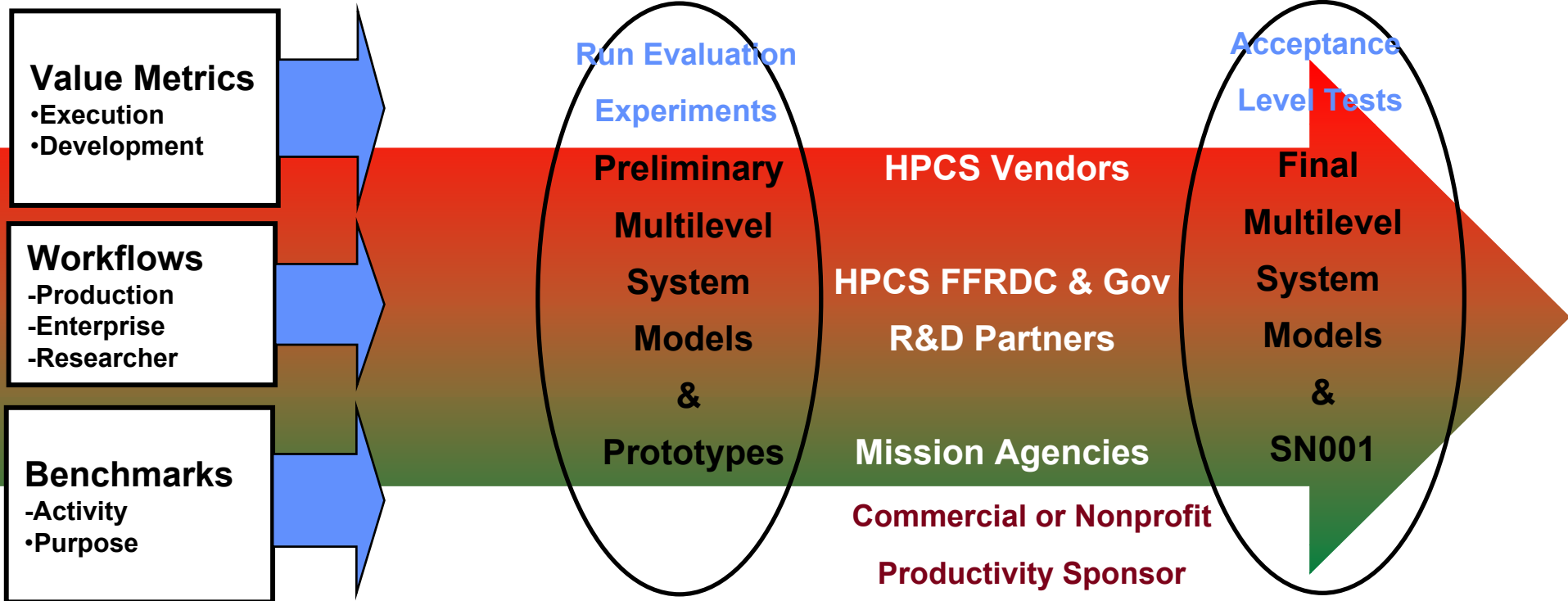
- Develop a procurement quality assessment methodology that will be the basis of 2010+ HPC procurements

# Productivity Framework Overview

**Phase I: Define Framework & Scope Petascale Requirements**

**Phase II: Implement Framework & Perform Design Assessments**

**Phase III: Transition To HPC Procurement Quality Framework**



**HPCS needs to develop a procurement quality assessment methodology that will be the basis of 2010+ HPC procurements**